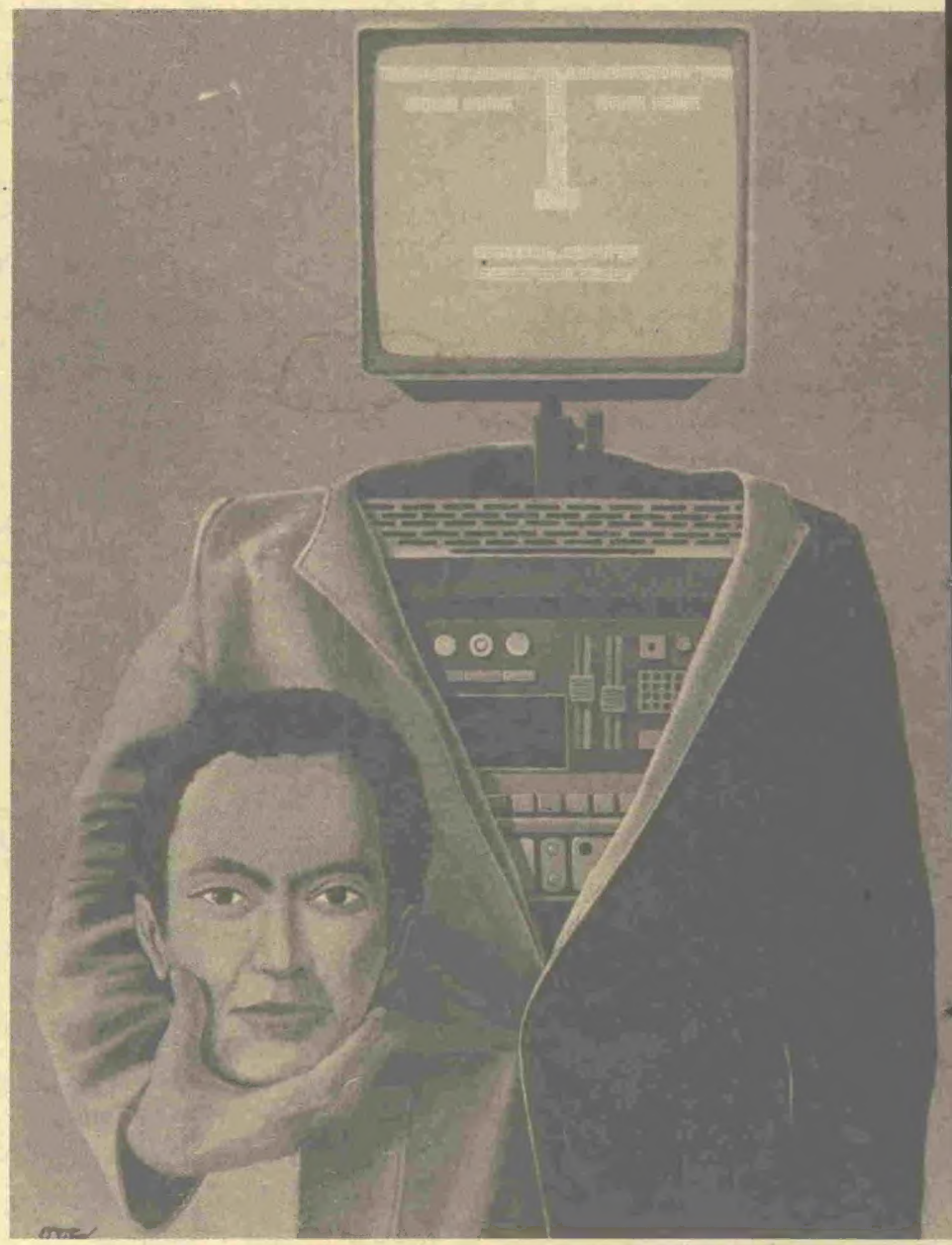
ยืน ภู่วรวรรณ
สมนึก คีรีโต

การประมวลผลภาษาธรรมชาติ

ความต้องการที่จะให้คอมพิวเตอร์เข้าใจภาษามนุษย์



ธรรมชาติสร้างสิ่งสลับซับซ้อนให้เกิดขึ้นและมีความสัมพันธ์กันอย่างแนบแน่น มนุษย์พยายามเรียนรู้กฎเกณฑ์ของธรรมชาติและหาทางนำมาใช้ประโยชน์ อย่างไรก็ตามก็ยังมีสิ่งต่าง ๆ ที่ธรรมชาติสร้างขึ้นที่มนุษย์ยังไม่สามารถหาคำอธิบายได้ ความสลับซับซ้อนของธรรมชาตินี้เองเป็นสิ่งที่ยากที่จะบรรยายหรือหากฎเกณฑ์มาอธิบายได้

ความรู้สึกนึกคิดต่าง ๆ ของมนุษย์ขึ้นอยู่กับอิทธิพลของธรรมชาติอยู่มาก ของสิ่งหนึ่งถ้าให้คนดูสองคน ความรู้สึกนึกคิดอาจแตกต่างกันออกไป ความรู้สึกของมนุษย์จึงเป็นกลไกที่ยากจะอธิบายได้ว่ามนุษย์คิดและเข้าใจได้อย่างไร

จุดเริ่มของความต้องการให้คอมพิวเตอร์เข้าใจภาษามนุษย์

ตั้งแต่คอมพิวเตอร์เครื่องแรกปรากฏโฉมขึ้นบนโลกโดยฝีมือนักประดิษฐ์เมื่อประมาณเกือบ 50 ปีที่แล้ว นักวิจัยในสมัยนั้นก็ได้พยายามหาทางให้คอมพิวเตอร์เข้าใจภาษามนุษย์ให้ได้ แนวความคิดของการสร้างโปรแกรมสั่งงานคอมพิวเตอร์ก็พยายามยึดเอาภาษาหรือคำศัพท์ที่มนุษย์เข้าใจได้ง่าย เริ่มจากการใช้ภาษาที่มีความหมายแทนภาษาเครื่องเช่น MOV A,B หรือ ADD พอจะบอกได้ว่าหมายถึงอะไร การนิยามภาษาที่จะมาใช้กับคอมพิวเตอร์ก็เกิดขึ้นและพัฒนาขึ้นมาเป็นลำดับ ดังจะเห็นได้จากภาษาที่มีการใช้กันมาถึงทุกวันนี้หลายภาษาได้นำเอาหลักการของภาษาธรรมชาติบางส่วนเข้าไปมีส่วนด้วย เช่น

```
LET A=5
LET B = A+8
PRINT A, B
```

การจัดการตัวแปลภาษาก็อาศัยโครงสร้างของภาษาธรรมชาติมาเป็นหลัก การสร้างกฎเกณฑ์ทางไวยากรณ์หรือ syntax เป็นความพยายามกำหนดกฎเกณฑ์ให้มีหลักการ เพื่อว่าตัวแปลภาษาจะได้จัดการทำงานอย่างถูกต้อง

งานที่คอมพิวเตอร์ทำได้มีเพียงน้อยนิด

หากจะพิจารณาดูว่าคอมพิวเตอร์เข้ามามีส่วนเกี่ยวข้องในการช่วยการ

ทำงานของมนุษย์เพียงไร เราจะพบว่างานที่คอมพิวเตอร์ทำได้มีเพียงน้อยนิด ซึ่งงานส่วนใหญ่เป็นงานที่มีรูปแบบแน่นอน (well form) และสามารถแทนข้อมูลได้ด้วยตัวเลข (numeric data) การวิจัยและพัฒนาในยุคหลังนี้จึงพยายามให้คอมพิวเตอร์เข้ามามีบทบาททางด้านการแก้ปัญหาที่มีรูปแบบไม่แน่นอนและมีโครงสร้างการให้เหตุผล (reasoning) ที่สลับซับซ้อนขึ้น เพื่อที่จะศึกษาพฤติกรรมความรอบรู้ของมนุษย์ การค้นคว้านี้จึงกลายมาเป็นศาสตร์ที่เรียกว่า **"ปัญญาประดิษฐ์"**

งานวิจัยจึงได้หาวิธีประยุกต์ใช้งานคอมพิวเตอร์ในรูปแบบอื่นเนื่องจากความต้องการของมนุษย์ต้องการเพิ่มประสิทธิภาพการทำงานในด้านต่าง ๆ ให้ดียิ่งขึ้น งานทางด้านปัญญาประดิษฐ์เกี่ยวพันและเป็นพื้นฐานของคนทุกคนที่มักจะเกี่ยวข้องด้วย หรือพูดให้ง่ายเข้าคือพยายามสร้างคอมพิวเตอร์ให้ใช้ง่าย ใครก็ได้สามารถใช้งานคอมพิวเตอร์ได้ และจะส่งผลทำให้คอมพิวเตอร์แพร่หลายและใช้งานกันมากขึ้น ซึ่งเป็นความต้องการของบริษัทผู้ผลิตอยู่แล้ว

ความต้องการในการสื่อสารระหว่างมนุษย์กับเครื่องจักรด้วยการใช้ภาษาธรรมชาติ

วิทยาการประมวลผลภาษาธรรมชาติได้ก้าวหน้าขึ้นเป็นลำดับ มนุษย์พยายามหาหนทางที่จะให้การติดต่อกันระหว่างมนุษย์กับเครื่องจักรด้วยการใช้ภาษาธรรมชาติ อย่างไรก็ตามรูปแบบทางภาษาธรรมชาติเป็นรูปแบบที่หาความแน่นอนได้ยาก เป็นศาสตร์ที่มีความสลับซับซ้อนมาก ในยุคแรกก่อนปี 2503 มนุษย์สร้างความหวังในเรื่องนี้และหลังจากปี 2503 ถึง 2513 เป็นยุคที่นักคอมพิวเตอร์ทั่วโลกกล่าวว่าการคิดค้นทางด้านภาษาธรรมชาติเป็นเรื่องที่เป็นไปไม่ได้เลย จนกระทั่งหลังปี 2513 ถึงปี 2520 เป็นยุคแห่งการเริ่มต้นค้นคิดทางด้านภาษาธรรมชาติกันใหม่ และในที่สุดเมื่อประมาณ 7-8 ปีมานี้มีผู้พยายามหาหนทางที่จะให้คอมพิวเตอร์เข้าใจภาษาธรรมชาติได้ดีขึ้น การค้นคิดจึงเริ่มผันแปรจากภาษาชนิด procedure language (เช่น โคบอล ปาสคาล ซี) มาเป็นภาษาแบบ declarative language เช่น PROLOG LISP การคิดค้นดูจะเห็นแนวทางแจ่มใสขึ้น

การจะให้คอมพิวเตอร์เข้าใจภาษาธรรมชาติได้ดีเพียงไรนั้นจึงขึ้นอยู่กับการพัฒนาสองแนวทาง แนวทางหนึ่ง คือ *พัฒนาการทางด้านปัญญาประดิษฐ์* ที่จะหาหนทางที่จะหาทางแทนความรอบรู้ (knowledge representation) และอีกแนวทางหนึ่งคือ *การศึกษาและเข้าใจโครงสร้างทางภาษาศาสตร์แบบมีโครงสร้างและเหตุผล* จุดเริ่มของการวิจัยทางภาษาศาสตร์ คือ ศาสตราจารย์ ชมสกี้ (chomsky) ได้คิดค้นสร้างทฤษฎี formal language ในส่วนของทฤษฎีนี้คือ context free (CF) มีบทบาทที่สำคัญมากต่อการพัฒนาการทางคอมพิวเตอร์และเป็นรากฐานของการพัฒนาตัวแปลภาษา (compiler และ interpreter) ในปัจจุบัน

ศาสตร์แห่งความสัมพันธ์ระหว่างสาขาภาษาศาสตร์ และคอมพิวเตอร์ศาสตร์จึงได้เกี่ยวโยงเชื่อมถึงกัน และกลายเป็นสาขาวิชาใหม่ขึ้นมาอีกหนึ่งสาขาวิชาคือ Computational Language (CL) เป็นวิชาการที่มีการศึกษากันจนถึงระดับปริญญาเอก CL และ AI จึงเจริญเติบโตมาด้วยกันจนก้าวเข้ามาสู่ยุคของ NLP ในปัจจุบัน

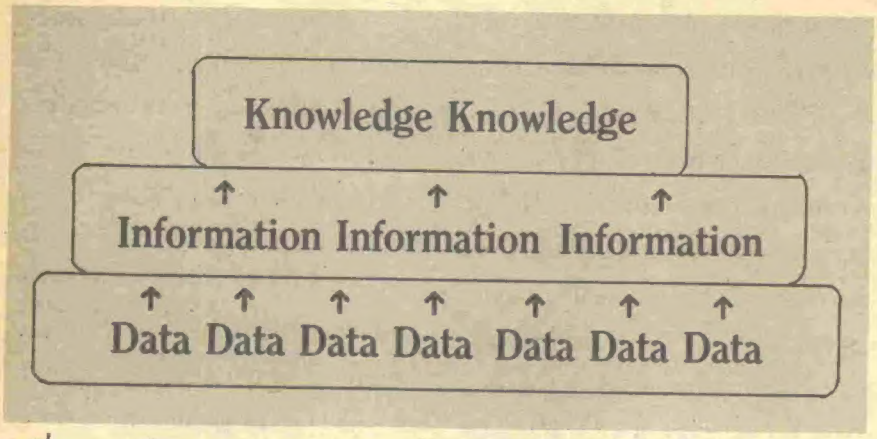
ก่อนจะมาเป็นข้อความที่จะใช้งานกับ NLP

การประมวลผลทางด้านคอมพิวเตอร์ได้แบ่งยุคสมัยออกมาเป็น 3 ยุค แล้ว โดยเริ่มจากยุคแรกตั้งแต่ยุคที่เริ่มมีคอมพิวเตอร์ใช้ เมื่อประมาณ 50 ปีที่แล้ว ยุคทั้งสามพอแบ่งได้ดังนี้

1. ยุคของการประมวลผลข้อมูล (Data Processing Age) เป็นยุคที่นำคอมพิวเตอร์มาคำนวณงานทางด้านวิทยาศาสตร์ เราใช้ภาษาคอมพิวเตอร์ช่วย เช่น ภาษาฟอร์แทรน โคบอล ฯลฯ การคำนวณจะกระทำเป็นงาน ๆ เครื่องคอมพิวเตอร์ในยุคนี้มีราคาแพงมาก การจัดการทางด้านข้อมูลส่วนใหญ่จึงเน้นในระบบ filing System คือนำข้อมูลจำนวนมากเก็บขึ้นเทป เมื่อต้องการใช้ก็นำข้อมูลจากเทปมาประมวลผล ระบบการทำงานลักษณะนี้จึงเรียกว่าการทำงานแบบ **เบตช์** ปัจจุบันยังมีศูนย์คอมพิวเตอร์ในประเทศไทยหลายแห่งยังใช้งานแบบนี้อยู่

2. ยุคของข้อมูลและข่าวสาร (Information Age) เป็นการมองข้อมูลเป็นระบบมากขึ้น การประมวลผลจะพิจารณาเป็นระบบโดยนำข้อมูลหลาย ๆ แฟ้มมาสัมพันธ์เกี่ยวข้องกันและเก็บรวบรวมไว้ในตัวกลางที่จะเรียกใช้ได้ตลอดเวลา ในยุคนี้จึงเกิดคำว่า **ดาต้าเบสหรือฐานข้อมูล** การทำงานของคอมพิวเตอร์จึงให้ความสำคัญของระบบที่เป็นระบบออนไลน์ ตัวอย่างระบบที่พบเห็นกันทั่วไป เช่น ระบบออนไลน์ของธนาคารที่ต้องมีฐานข้อมูลกลาง ระบบการให้บริการข่าวสารข้อมูลจากฐานข้อมูลเฉพาะ เป็นต้น ปัจจุบันการออกแบบระบบมักเน้นในระบบฐานข้อมูล ดังนั้น เราจึงมีตัวช่วยจัดการฐานข้อมูลซึ่งเป็นซอฟต์แวร์ออกมาเป็นจำนวนมาก มาตรฐานของการจัดการฐานข้อมูลก็มีหลายรูปแบบซึ่งเหมาะกับระบบงานที่แตกต่างกันออกไป

3. ยุคของการประมวลความรอบรู้ (Knowledge Processing Age) เป็น



รูปที่ 1 แสดงโมเดลของการประมวลผลด้วยคอมพิวเตอร์

ยุคของคอมพิวเตอร์ที่เรากำลังก้าวมาถึง การคืนตัวของเทคโนโลยี AI ทำให้เรากำลังหาหนทางนำเอาความรอบรู้ มาใช้ประโยชน์ให้มากขึ้น ระบบทางซอฟต์แวร์ที่กำลังได้รับการกล่าวขวัญในยุคนี้ คือ **ระบบผู้ชำนาญการ (expert system) เรื่องของ ฐานความรอบรู้ (knowledge base)** หรือแม้กระทั่งระบบช่วยให้คำปรึกษาและตัดสินใจต่าง ๆ การประมวลผลในยุคนี้จึงเกี่ยวข้องกับการใช้ภาษาธรรมชาติมากยิ่งขึ้น ระบบการใช้ฐานความรอบรู้นี้จึงเป็นศาสตร์ที่นำเอาเทคนิคการแทนความรอบรู้มาใช้ และด้วยเทคนิควิธีการนี้เองก็ได้ถูกนำมาประสานเข้ากับระบบการแทนความรอบรู้ทางภาษาธรรมชาติเพื่อให้คอมพิวเตอร์เข้าใจภาษามนุษย์ได้ดีขึ้นนั่นเอง

ศึกษาโครงสร้างของตัวอักขระที่ใช้ในคอมพิวเตอร์ที่ทั่วโลกใช้

ภาษาธรรมชาติเป็นภาษาที่มีวิวัฒนาการมาเป็นระยะเวลายาวนานหลายพันปี ภาษาธรรมชาติในแต่ละภาษาจะมีลักษณะเฉพาะตัวของตนเอง มีการเรียงรูปประโยคที่แตกต่างกันออกไป เพื่อให้เห็นภาพของการประมวลผลแบบ NLP ชัดเจนยิ่งขึ้น ผู้เขียนขอเสนอแนวความคิดบางอย่าง และจะเน้นเฉพาะในบางเรื่องที่เป็นภาษาไทย เพื่อเป็นแนวทางให้เห็นภาพของการพัฒนา NLP ของเราต่อไปในระยะยาว

ก่อนอื่นผู้เขียนอยากเน้นให้เห็นโครงร่างของตัวอักขระที่ใช้กันอยู่ทั่วโลกก่อน ในแต่ละชนชาติจะมีภาษาเป็นของตนเองเพื่อแสดงเอกลักษณ์และความเป็นอิสระของแต่ละประเทศ รูปแบบโครงร่างของตัวอักขระที่จะพิจารณาจะแบ่งแยกออกเป็น 2 กลุ่มใหญ่ คือ กลุ่มที่เป็นตัวอักขระและผสมเป็นคำ และกลุ่มที่เป็นอักขระในลักษณะรูปภาพ

กลุ่มที่เป็นตัวอักขระพื้นฐาน เป็นกลุ่มที่ใช้กันมากในทั่วโลก อักขระเหล่านี้จะแทนการออกเสียงและนำมาผสมกันเป็นคำหรือข้อความ เช่น อักขระลาตินที่ใช้ในภาษาอังกฤษ-อเมริกัน จะมีเพียง 26 ตัวเท่านั้นแต่ผสมกันเป็นข้อความหรือประโยคตามที่ต้องการ กลุ่มอักขระนี้จะแทนด้วยตัวเลขที่เป็นมาตรฐานเช่น รหัส EBCDIC, ASCII อักขระเหล่านี้อาจจะเป็นอักขระตัวอักษร ตัวเลข และเครื่องหมายพิเศษ ตามมาตรฐาน ASCII จะมีตัวอักษรเพียง 96 ตัวที่บรรจุในตาราง ASCII อย่างไรก็ตามเมื่อใช้ในบางประเทศ เช่น เยอรมันนี อิตาลี หรือบางประเทศทางยุโรป ก็จะมีอักขระพิเศษเพิ่มเติมออกไป ประเทศที่ใช้ตัวอักขระพื้นฐานที่แทนเสียงนับว่าเป็นการโชคดีที่จะใช้กับคอมพิวเตอร์ เพราะกรรมวิธีการประมวลผล การป้อนข้อมูลหรือการแสดงผลทำได้ง่าย ดังนั้น จึงมีบางประเทศ เช่น มาเลเซีย อินโดนีเซีย ใช้หลักการของการโรมาไนซ์เพื่อให้ได้เสียงตามที่ต้องการโดยใช้อักขระในตาราง ASCII ได้โดยตรง

กลุ่มตัวอักขระที่ใช้แบบรูปภาพ ซึ่งอาจจะกล่าวได้ว่าแต่ละคำจะมีรูปแบบของตัวอักขระเฉพาะ ดังนั้นในภาษาธรรมชาติในกลุ่มนี้จะมีตัวอักขระมากกว่า 50,000 ตัว กลุ่มตัวอักขระเหล่านี้ได้แก่ประเทศที่อยู่ทางเอเชียตะวันออก เช่น จีน ญี่ปุ่น เกาหลี ความพยายามในการประมวลผลกลุ่มภาษานี้ก็มีมาพร้อมกับยุคคอมพิวเตอร์เช่นกัน

ตัวอักษรจีนเป็นตัวอักษรที่มีวิวัฒนาการมากว่า 4000 ปีแล้ว และมีหลักฐานที่เชื่อได้ว่าตัวอักษรจีนที่ใช้กันในปัจจุบันนี้ได้ใช้กันมานานกว่า 2000 ปี ตัวอักษรจีนที่ใช้กันนี้เรียกชื่อว่า hanzi และเมื่อมาใช้ในญี่ปุ่นตัวอักษรที่เป็นแบบรูปภาพนี้จะเรียกว่า kanji และเมื่อใช้ในเกาหลีก็จะเรียกว่า haja

ตัวอักษร hanzi แต่ละตัวจะแทนคำพูดหนึ่งพยางค์ ส่วนภาษาเกาหลีและญี่ปุ่นเป็นภาษาที่ยืมตัวอักษร hanzi มาเมื่อประมาณ 1500 ปีที่แล้วและพัฒนาการแตกต่างออกมาทั้งสำเนียงและการออกเสียง แต่ก็ได้มีการพัฒนาการทางด้านไวยากรณ์ของภาษาที่แตกต่างกันออกไปอีก นอกจากนี้ทั้งญี่ปุ่นและเกาหลีได้มีการพัฒนาตัวอักษรที่ใช้โทนเสียงแบบ phonetic เพื่อที่จะแทนคำที่ยืมมาจากภาษาต่างประเทศให้คล่องตัวขึ้น ในกรณีของญี่ปุ่นจึงมีตัวอักษรที่เรียกว่า hiragana และ katakana ส่วนของเกาหลีมีตัวอักษรที่เรียกว่า hangul ตัวอักษรที่ใช้ในการออกเสียงนี้ทั้งญี่ปุ่นและเกาหลีได้นำมาใช้กว้างขวางขึ้นในช่วงร้อยปีนี้เอง ภาษาญี่ปุ่นจึงเป็นภาษาผสมที่

ใช้ตัวอักษรที่มาจาก hanzi, hiragana, katakana

ความเกี่ยวพันของตัวอักขระที่ใช้ในภาษาจีน ญี่ปุ่นและเกาหลี

ผู้เขียนใคร่อยากให้เห็นลักษณะวิวัฒนาการทางภาษาและความสัมพันธ์ของแต่ละภาษาซึ่งถึงแม้ว่าจะใกล้เคียงกันก็ยังมีสิ่งที่แตกต่างกันอยู่มาก การศึกษาภาษาแต่ละภาษานั้นจึงเป็นเรื่องเฉพาะและผู้ที่เป็นเจ้าของภาษาจะเป็นผู้ที่รู้ดีที่สุด การให้คอมพิวเตอร์มาช่วยประมวลผลภาษานั้นเป็นความต้องการในยุคปัจจุบัน

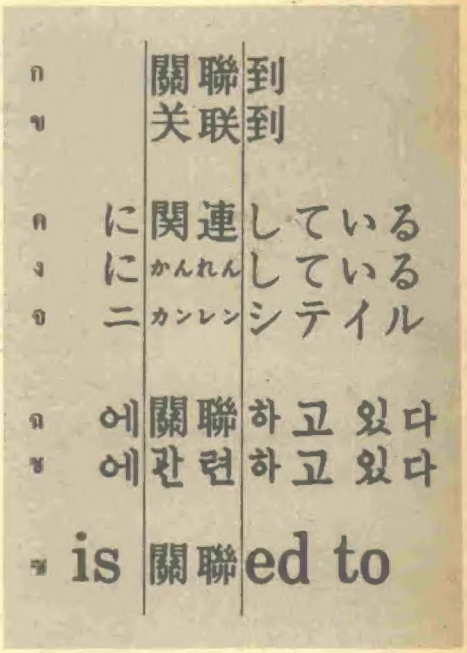
มีคำถามถามว่าจำนวนตัวอักขระหรือคำเท่าไรในภาษาจีนจึงจะพอใช้ในการพิมพ์เอกสารเช่นเวิร์ดโปรเซสเซอร์ การที่ศึกษาจำนวนคำหรือลักษณะรูปแบบการใช้คำนี้มีความจำเป็นและการดูตัวอย่างของภาษาต่างประเทศจะทำให้เราได้แนวทางมาใช้กับภาษาไทยได้ดียิ่งขึ้น

ปราชญ์ผู้เชี่ยวชาญทางภาษาจีนได้กล่าวไว้ว่าจำนวนตัวอักษรภาษาจีนที่ใช้กันถ้าจะให้ได้ในระดับที่ประทับใจหรือมีตัวอักษรให้ใช้กันอย่างพอเพียงแล้วจะตกประมาณ 50,000 ตัว (ความเป็นมาที่ใช้รหัส 2 ไบต์ ที่ผู้เขียนอยากจะกล่าวต่อไปในโอกาสหน้า) ดังนั้น ดิกชันนารีที่ใช้กับ hanzi ในการเขียนจะมีอยู่ประมาณ 50,000 ตัวดังที่กล่าวแล้ว

แต่ในทางปฏิบัติเมื่อนำมาใช้ในงานเวิร์ดโปรเซสเซอร์ภาษาจีน การให้คอมพิวเตอร์มองหาคำในดิกชันนารีมาใช้ จะใช้เพียงประมาณ 7,000 คำ (ตรงกับผู้เขียนที่เคยวิเคราะห์คำไทยที่ใช้บ่อยในชีวิตประจำวันจะอยู่ในราว 7,000 คำเช่นกัน ดังนั้นดิกชันนารีที่ใช้แบ่งคำไทยจะมีเพียง 7,000 คำ ก็ช่วยได้มาก) แต่ในระดับที่มีผู้ผลิตเวิร์ดโปรเซสเซอร์ภาษาจีนที่ขายกันในท้องตลาดจะใช้น้อยกว่า 7,000 คำด้วย คือใช้คำหลักเพียง 4,000 คำ และกรณีเวิร์ดโปรเซสเซอร์ภาษาญี่ปุ่นจะใช้ kanji ประมาณ 2,000 คำ อย่างไรก็ตามสำหรับประเทศเกาหลีนั้นปัจจุบันหันมาใช้ hangul มากกว่าที่จะใช้ตัวอักษรจีน hanzi

เพื่อให้เห็นแนวทางความสัมพันธ์ของสามภาษานี้ จึงขอให้ดูคำศัพท์ของภาษาทั้ง 3 ที่มีความสัมพันธ์กัน ซึ่งในภาษาจีนก็เหมือนกับภาษาธรรมชาติอื่นที่ต้องอาศัยคำผสมเพื่อสร้างคำใหม่ เช่น ในภาษาไทย *ไฟฟ้า* จะหมายถึง fire, sky เมื่อรวมกันเป็น electric hanzi ก็เช่นกันนำคำสองคำมาสร้างความหมายใหม่ได้เช่นกัน

ลองพิจารณาคำในรูปที่ 2 ที่ขีดเส้นคร่อมลงมาซึ่งมีความหมายเหมือนกัน ในคำแรกของจีนอ่านว่า guan และคำที่สองอ่านว่า lian เมื่อมารวมกันอ่านว่า guan-lian ซึ่งมีความหมายว่า relate การเพิ่มคำที่สามเข้ามาอ่านว่า guan-lain dao ซึ่งมีความหมายเท่ากับ is related to หรือ "สัมพันธ์กับ" ในแถว ก. และ ข.เป็นภาษาจีน ในแถว ก. เป็นภาษาจีนที่ใช้มาแต่โบราณกาล ในแถว ข. ได้ปรับปรุงมาเมื่อปี พ.ศ. 2498 เพื่อให้เขียนง่าย



รูปที่ 2 ความแตกต่างของการใช้ภาษาในกลุ่มภาษาเดียวกัน

รูปที่ 3 ส่วนหนึ่งของตารางรหัส 2 ไบต์ ตามมาตรฐานประเทศญี่ปุ่นสังเกตการจัดกลุ่มจะจัดตามกลุ่มเสียง

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
力	3230													下	化	仮	何
	3240	伽	価	佳	加	可	嘉	夏	嫁	家	寡	科	暇	果	架	歌	河
	3250	火	珂	禍	禾	稼	箇	花	苛	茄	荷	華	菓	蝦	課	嘩	貨
	3260	迦	過	霞	蚊	俄	峨	我	牙	画	臥	芽	蛾	賀	雅	餓	駕
	3270	介	会	解	回	塊	壊	廻	快	怪	悔	恢	懐	戒	拐	改	
	3320		魁	晦	械	海	灰	界	皆	絵	芥	蟹	開	階	貝	凱	劾
	3330	外	咳	害	崖	慨	概	涯	碍	蓋	街	該	鎧	骸	浬	馨	蛙
	3340	垣	柿	蛎	鈎	劃	嚇	各	廓	拡	攪	格	核	殻	獲	確	穫
	3350	覚	角	赫	較	郭	閣	隔	革	学	岳	楽	額	顎	掛	笠	樫
	3360	橿	梶	鰍	潟	割	喝	恰	括	活	渇	滑	葛	褐	轄	且	鰹
	3370	叶	椛	樺	鞄	株	兜	竈	蒲	釜	鎌	噛	鴨	栢	茅	萱	
	3420		粥	刈	苅	瓦	乾	侃	冠	寒	刊	勘	勧	巻	喚	堪	姦
	3430	完	官	寛	干	幹	患	感	慣	憾	換	敢	柑	桓	棺	款	歓
	3440	汗	漢	澗	灌	環	甘	監	看	竿	管	簡	緩	缶	翰	肝	艦
	3450	莞	観	諌	貫	還	鑑	間	閑	関	陥	韓	館	舘	丸	含	岸
	3460	巌	玩	癌	眼	岩	翫	贋	雁	頑	顔	願					

ตัวอักขระบนคีย์บอร์ด	ข้อความที่แสดงบนจอ	ลำดับการกดคีย์
ㅎ	ㅎ	ㅎ
ㅏ	하	ㅎ ㅏ
ㄴ	하ㄴ	ㅎ ㅏ ㄴ
ㄱ	한ㄱ	ㅎ ㅏ ㄴ ㄱ
ㅜ	한구	ㅎ ㅏ ㄴ ㄱ ㅜ
ㄱ	한구ㄱ	ㅎ ㅏ ㄴ ㄱ ㅜ ㄱ
ㅇ	한국ㅇ	ㅎ ㅏ ㄴ ㄱ ㅜ ㄱ ㅇ
ㅓ	한국어	ㅎ ㅏ ㄴ ㄱ ㅜ ㄱ ㅇ ㅓ

รูปที่ 4 ตัวอย่างการพิมพ์ข้อความตัวอักษรเกาหลีจากอักขระ hanzi

เข้าแต่ก็ยังคงอ่านเหมือนเดิม ครั้นมาใช้เป็นภาษาญี่ปุ่น จะเขียนได้ดังแถว ค.ง. และจ. แถวค.เป็นการเขียนในรูปแบบ kanji อ่านว่า "kan-ren" ซึ่งมีสำเนียงเพี้ยนไปจากจีน ตัว kanji ก็ได้ปรับปรุงเปลี่ยนแปลงไปจากตัว hanzi เดิม ส่วนในแถว ง. เป็นการเขียนในรูปแบบ hiragana เพื่อออกเสียงเหมือนเดิมแต่ก็ไม่ค่อยใช้นัก ในแถว จ. เป็นการเขียนในรูปแบบ katakana ซึ่งก็ไม่ค่อยใช้กับคำนี้ในภาษาญี่ปุ่น เพราะจุดมุ่งหมายของ katakana นั้นไว้สำหรับการทับศัพท์ที่มาจากภาษาต่างประเทศ

หมายเหตุผู้เขียน

ผู้เขียนมิใช่เป็นผู้ที่สันทัดภาษาญี่ปุ่น เกาหลี หรือจีน แต่ก็พยายามศึกษารูปแบบในการศึกษาแบบทางคอมพิวเตอร์ เพื่อเป็นแม่แบบของการเปรียบเทียบกับภาษาไทย อนึ่งทีมงานวิจัยในห้องแลบที่ผู้เขียนทำงานอยู่ได้สนใจเรื่องการแปลภาษาด้วยเครื่อง ดังนั้น ข้อความและเอกสารจึงต้องอ้างอิงกับของต่างประเทศหลาย ๆ ภาษาเพื่อการศึกษา

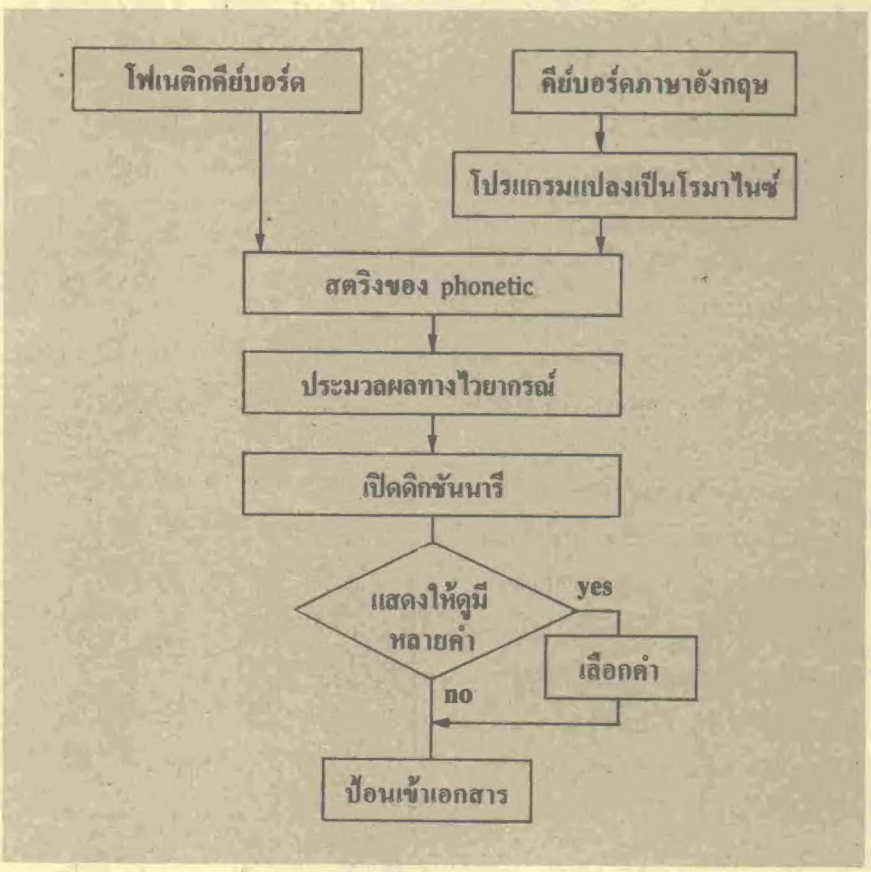
การยกตัวอย่างในบทความนี้เป็นเพียงให้เห็นรูปแบบของภาษาธรรมชาติในกลุ่ม picture character ซึ่งเป็นอีกกลุ่มหนึ่งที่เราจะต้องให้ความสนใจเป็นพิเศษด้วย เพราะด้วยการเจริญก้าวหน้าของคอมพิวเตอร์เจริญมาจากด้านทางตะวันออกและตะวันตกซึ่งมีรูปแบบบางอย่างแตกต่างกัน

ส่วนในแถว ฉ. และช. เป็นการเขียนในรูปแบบภาษาเกาหลี แต่ภาษาเกาหลีอ่านออกเสียงเพี้ยนไปเป็น kwal-lyen แถว ฉ. เป็นรูปแบบที่ภาษาเกาหลีใช้ แต่ในยุคปัจจุบันมักนิยมนำเอา hangul มาใช้ดังที่เห็นในแถว ช.

สำหรับแถวสุดท้ายให้เห็นคำศัพท์ในภาษาอังกฤษที่ใช้ในความหมายเดียวกัน สังเกตการใช้รูปไวยากรณ์ของแต่ละภาษาจะมีรูปแบบตัวช่วย เช่น is, to หรือคำสันธานที่ใช้เชื่อมในรูปประโยคเพื่อให้ถูกต้องตามไวยากรณ์

การประมวลผลกลุ่มภาษาจีนยังมีปัญหาที่จะเอาชนะอีกมาก

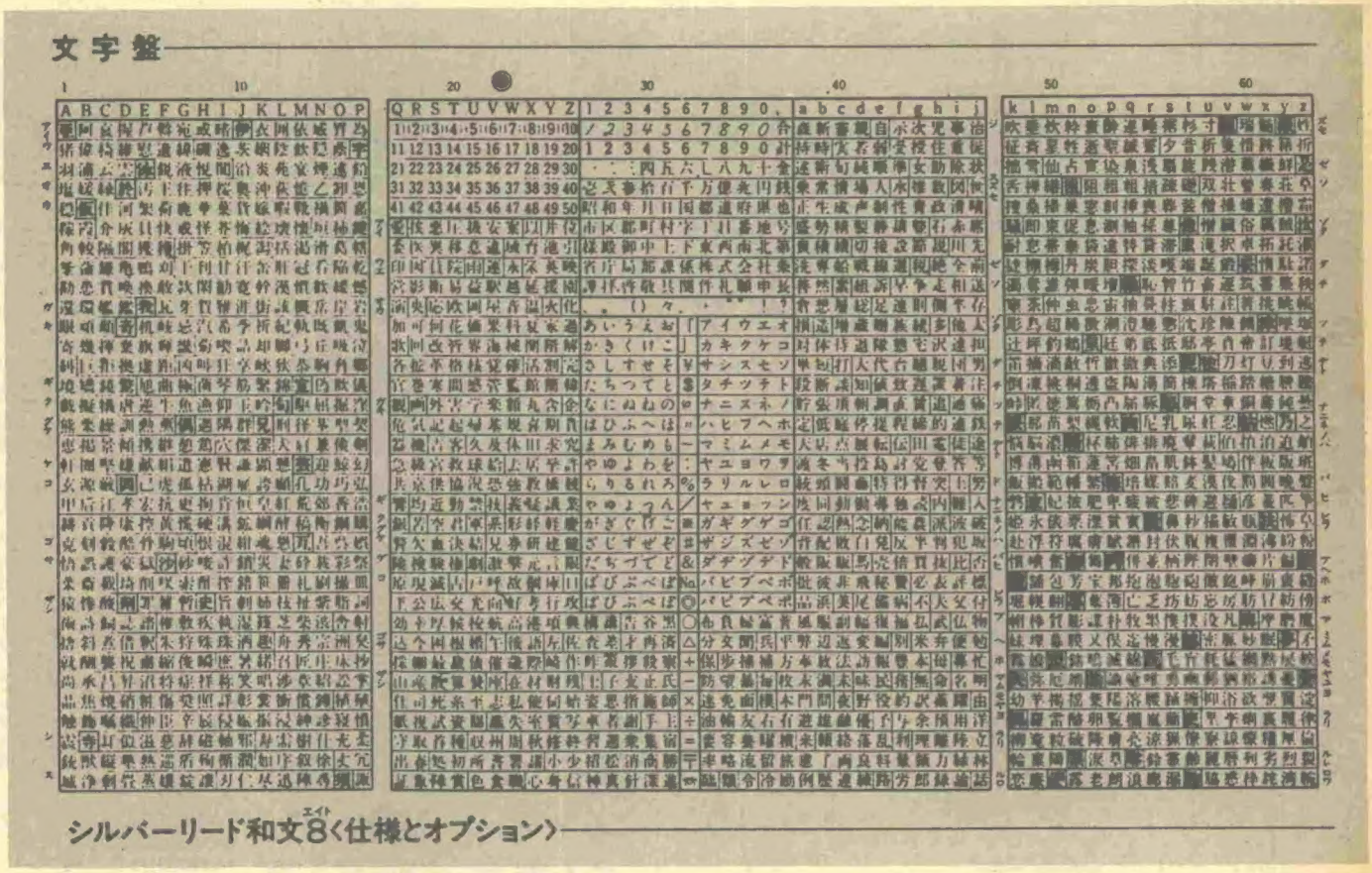
ในบรรดาภาษาทั้งสามคือ จีน ญี่ปุ่น และเกาหลี ล้วนมีลักษณะเฉพาะตัวของตัวเองทั้งสิ้น ในกลุ่มของภาษาเกาหลีนั้น ตัวอักขระ hangul เพิ่งได้รับการ



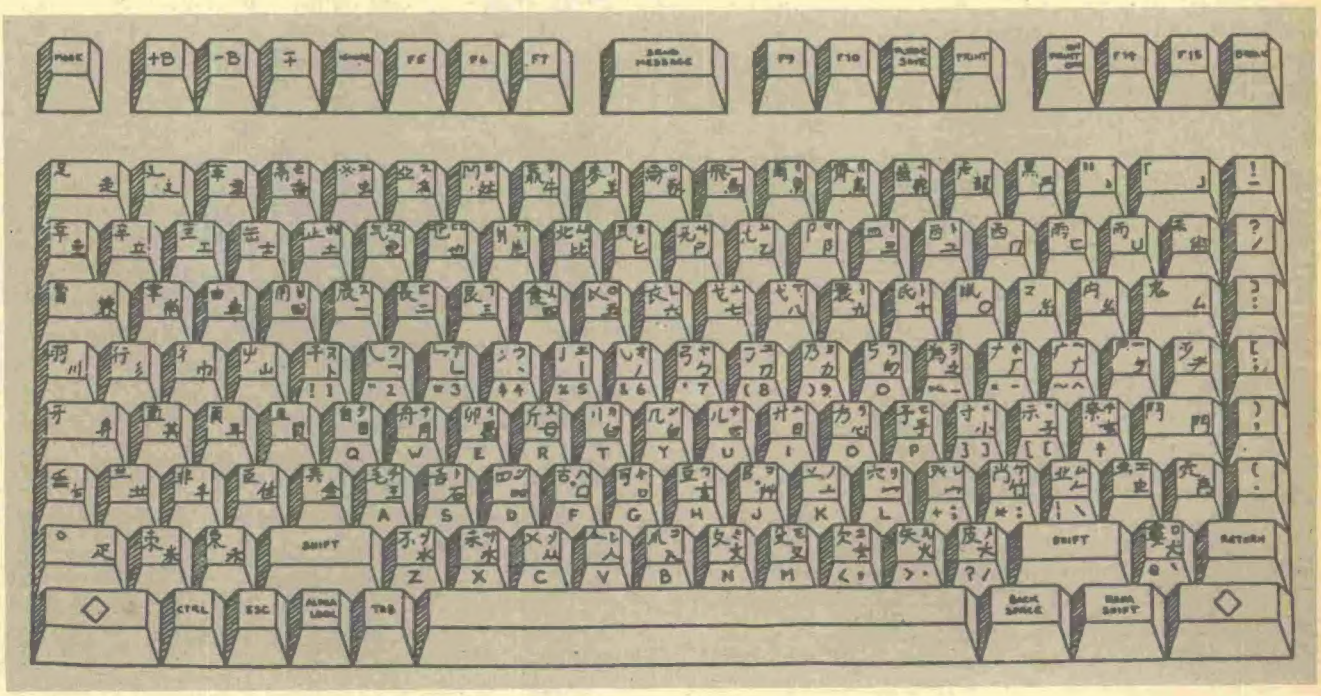
รูปที่ 5 ลำดับการใช้แบบ phonetic คีย์บอร์ด

ออกแบบและกำหนดขึ้นเมื่อปีพ.ศ. 1986 หรือประมาณ 500 กว่าปีมาแล้ว ตัวอักษร hangul 40 ตัว สามารถนำมาประกอบเป็นตัวอักษรที่มีความหมาย 500 ตัว และรวมกันเป็นพยางค์ของคำที่ใช้ได้ถึงประมาณ 2,000 คำ การอินพุตทางคีย์บอร์ดของภาษาทั้งสามจึงต้องมีลักษณะที่ยุ่งยากกว่าภาษาทางตะวันตกที่ใช้อักขระโรมันมาก

วิธีการป้อนข้อมูลภาษาในกลุ่ม hanzi ได้พัฒนากันมาตั้งแต่ พ.ศ. 2456 ในยุคของการมีเครื่องพิมพ์ดีด การพัฒนา



รูปที่ 5 ข. ตาราง 2 มิติสำหรับอินพุตตัวอักษร KANJI



รูปที่ 5 ค. ลักษณะคีย์บอร์ดที่คีย์เฉพาะส่วนของตัวอักษร

ค่อย ๆ ก้าวหน้าขึ้นเป็นลำดับจากยุคเครื่องพิมพ์ดีดกลไกมาเป็นเครื่องพิมพ์ดีดไฟฟ้าและกลายมาเป็นเวิร์ดโปรเซสเซอร์ วิธีการป้อนข้อมูลที่ทำกันมาจนถึงปัจจุบันนี้แบ่งออกเป็น 4 วิธี คือ

● **วิธีการเลือกจากตาราง 2 มิติ** เป็นการกำหนดตัวอักษรลงไปในตาราง 2 มิติ ซึ่งสามารถกำหนดตัวอักษรได้เป็นพัน ๆ ตัว การป้อนข้อมูลนี้จะใช้ปากกาจี้ลงไปที่ตำแหน่งของตัวอักษรที่ต้องการ หรือการบอกตำแหน่งแถวและคอลัมน์ ดังตัวอย่างในรูป 5 ข.

● **การใช้รหัสตามลักษณะรูปร่างที่ผสมกัน** เช่น ตัวอักษรจีนหนึ่งตัวมีส่วนประกอบของรูปร่างส่วนย่อยหลาย ๆ ส่วน เราแบ่งแยกแล้วนำมาป้อน โดยกำหนดคีย์บอร์ดที่มีตัวอักษรของรูปร่างส่วนย่อยต่าง ๆ เหล่านั้นไว้

● **การใช้คีย์ 2 คีย์ ผสมเป็นตัวอักษร** วิธีนี้ผู้ใช้อาจยุ่งยากเพราะต้องจำให้ได้ว่าตัวอักษรแต่ละตัวเกิดจากคีย์ผสมระหว่างคีย์อะไรกับอะไร

● **การใช้หลักการออกเสียง** กรณีนี้อาจใช้คีย์บอร์ดภาษาอังกฤษคีย์ตามเสียงและโปรแกรมคอมพิวเตอร์จะมองหาจากดิกชันนารีมาแสดงให้

ภาษาไทย : เทคโนโลยีที่ต้องพึ่งพิงทั้งทางตะวันออก-ตก

ประเทศไทยมีการนำเข้าคอมพิวเตอร์ตั้งแต่เครื่องแรกในปี พ.ศ. 2507 และจนบัดนี้เลยมายี่สิบกว่าปีแล้ว เราก็ยังต้องนำเข้าจากต่างประเทศ และที่สำคัญคือเราต้องนำเข้าทั้งจากประเทศทางด้านตะวันตกคือยุโรปและอเมริกา และทางตะวันออกคือ ญี่ปุ่น เกาหลี ไต้หวัน (จีน) การศึกษาแนวทางการพัฒนาเทคโนโลยีทั้งสองค่ายที่จะมีอิทธิพลต่อการใช้งานในประเทศจึงเป็นสิ่งที่ต้องกระทำ

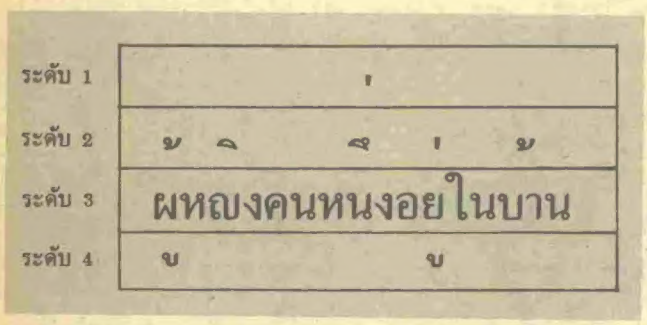
โดยธรรมชาติโครงสร้างภาษาไทยมีลักษณะเฉพาะตัว คือ เมื่อมองแบบเป็นตัวอักษร เราสามารถใช้ในระบบพิมพ์ดีด และที่สำคัญคือ เรามีอักขระที่บรรจุลงในตารางได้ไม่เกิน 96 ตัว ซึ่งสามารถบรรจุลงในตารางรหัส 8 บิต ใช้ร่วมกับรหัส ASCII หรือ EBCDIC ได้ เรามีคณะกรรมการจัดการสร้างรหัสมาตรฐาน สมอ. (แต่ไม่ได้มีการดำเนินการแบบต่อเนื่อง ผลของรหัสสมอ.ที่ออกมาจึงเหมือนเด็กพิการที่ไม่รู้จักโต) แต่เพียงรหัสสมอ. นั้นน่าจะขยายระดับเชื่อมต่อเข้าสู่มาตรฐานสากลได้ทั้งทางค่ายตะวันตกและตะวันออก

สำหรับในฉบับนี้ผู้เขียนอยากเสนอแนวทางความคิดที่เราน่าจะเดินทางเพื่อดึงเทคโนโลยีทั้งทางตะวันออกและตะวันตกมาใช้ให้เกิดประโยชน์กับเรามากที่สุด ผู้เขียนขอให้พิจารณาโครงสร้างภาษาไทยแบบพื้นฐานก่อน

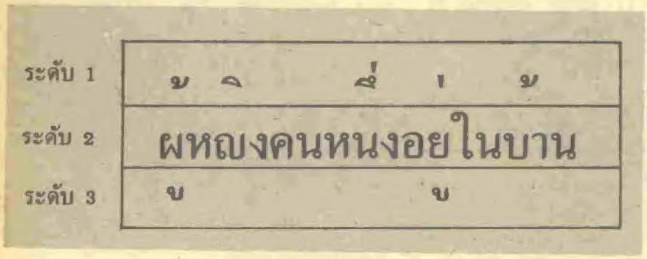
ภาษาไทยประกอบด้วย พยัญชนะ สระ วรรณยุกต์ เครื่องหมายพิเศษ ซึ่งมีลักษณะเฉพาะคือการวางระดับออกเป็น 4 ระดับ ดังรูปที่ 6

โครงสร้างภาษาไทยจัดแบบ hanzi ได้

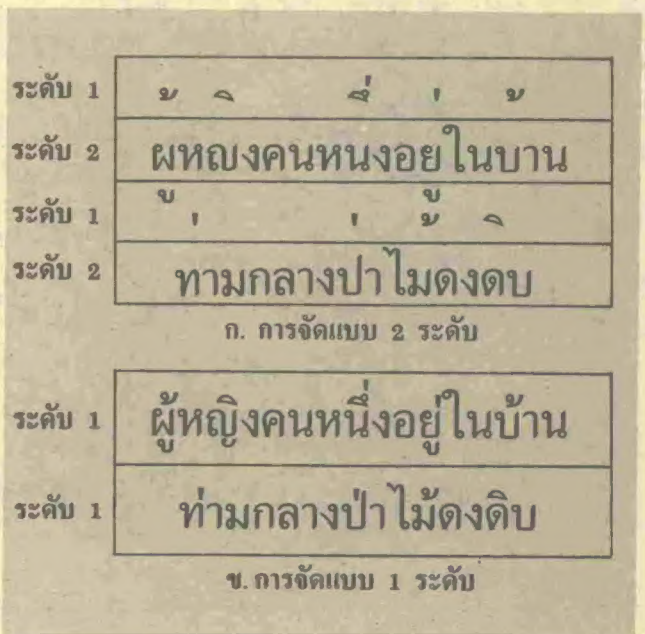
การพัฒนาภาษาไทยในยุคแรกเน้นการพัฒนาทางด้านอินพุตเอาต์พุต เพื่อให้โปรแกรมสำเร็จรูปใช้กับข้อมูลภาษาไทยได้ หรือสามารถป้อนข้อมูลภาษาไทยได้ และด้วยเหตุผลของการพัฒนาทางด้านอินพุตเอาต์พุต เราจะพบปัญหาว่าโครงสร้างอักษรโรมันกับภาษาไทยมีข้อแตกต่างกันมากในการประมวลผล เพราะ



รูปที่ 6 การแบ่งระดับของภาษาไทย



รูปที่ 7 การจัดแบบ 3 ระดับที่เครื่องพิมพ์ใช้งานส่วนใหญ่



รูปที่ 8 การลดระดับเหลือ 1 ระดับหรือ 2 ระดับ

ภาษาธรรมชาติกับการแทนด้วยคณิตศาสตร์ลอจิก

ความพยายามที่จะหาทางแก้ปัญหาทางลอจิกมีมาตั้งแต่สมัยกรีกโบราณ การแก้ปัญหาทางลอจิกเป็นปัญหาที่พบเห็นกันอยู่ในชีวิตประจำวัน ในที่สุดก็เกิดคณิตศาสตร์ขึ้นมาแขนงหนึ่ง เป็นคณิตศาสตร์ที่มีรากของการพัฒนาคอมพิวเตอร์ในปัจจุบัน คณิตศาสตร์แขนงนั้นคือ "คณิตศาสตร์ลอจิก" (Mathematical Logic)

คณิตศาสตร์ลอจิกได้เข้ามามีบทบาทต่อการแก้ปัญหาหลายสิ่งหลายอย่างโดยเฉพาะปัญหาที่มนุษย์ต้องการเหตุและผล ในที่สุดก็เกิดพีชคณิตขึ้นมาที่เรียกว่า Boolean Algebra และต่อมาได้กลายมาเป็นต้นกำเนิดของการศึกษาเกี่ยวกับการจัดการทางเหตุผลที่ทำต่อประโยคที่เราเรียกว่า Propositional logic

Propositional logic ได้เข้ามามีบทบาทในการแทนปัญหาทางลอจิก เช่น เมื่อมีปัญหาทางลอจิกแล้วเราจะแทนปัญหานั้นได้อย่างไร และมีวิธีการแก้ปัญหาอย่างไร เช่นปัญหา

"มานะ มานี และ วีระ ทานอาหารเย็นด้วยกันทุกวัน

ถ้ามานะสั่งข้าวผัด มานีจะต้องสั่งราดหน้า

ระหว่างมานะกับวีระต้องมีใครสักคนหนึ่งสั่งข้าวผัดและอีกคนสั่งราดหน้า แต่มานีและวีระไม่เคยกินราดหน้าพร้อมกันเลย คำถามมีว่า ใครทานอะไรบ้างในแต่ละวัน"

ปัญหาดังกล่าวนี้เป็นปัญหาเชิงลอจิกและการแก้ปัญหาได้ด้วยคณิตศาสตร์ลอจิกเช่นกัน

คณิตศาสตร์ลอจิกเน้นการถ่ายทอดความจริง

การถ่ายทอดความจริง (inference) เป็นกลไกของเหตุผลที่อาศัยคณิตศาสตร์ลอจิกเข้าช่วย โดยปกติการถ่ายทอดความจริงจะมาจากการตั้งสมมติฐาน และส่งผลไปยังข้อสรุป ดังนี้

Hypothesis → Conclusion

เช่นการกล่าวว่า เมื่อฝนตกกบจะร้อง จะเห็นว่า ประกอบด้วยการถ่ายทอดความรู้ในลักษณะเหตุผล คือ

ฝนตก เป็น Hypothesis

ภาษาไทยมีการจัดระดับอักขระบนล่าง ทำให้รูปแบบการเก็บข้อมูลทาง physical กับรูปแบบการแสดงผลซึ่งเป็นแบบ 3-4 ระดับไม่ตรงกัน ดังนั้นโปรแกรมสำเร็จรูปจากต่างประเทศจึงยากที่จะทำให้สมบูรณ์แบบได้ เช่น เซลโลตัสในภาวะปกติ แสดงได้ 9 ตัวอักษร แต่เมื่อแสดงอักขระไทยที่จัดแบบ 4 ระดับจะแสดงได้ไม่ครบ 9 คอลัมน์ เพราะจะแสดงได้ 9 ตัวอักษร (9 ตัวอักษรภาษาไทยไม่จำเป็นต้องแสดงผล 9 คอลัมน์)

การแสดงผลภาษาไทยเพื่อความสวยงามจึงจัดระดับด้วยตัวอักษรผสมลดลงเหลือสามระดับดังรูปที่ 7

อย่างไรก็ดียังมีผู้พัฒนาภาษาไทยแบบ 25 บรรทัด เพื่อแสดงผลบนจอภาพขนาด 12" หรือ 14" ได้จึงต้องหาวิธีขยายขนาดตัวอักษรและลดช่องว่างระหว่างบรรทัด ทำให้การจัดระดับตัวอักษรเป็นแบบ 2 และ 1 ระดับได้ ดังรูปที่ 8

ภาษาไทย : รหัสที่น่าจะมีหลายชั้น

รหัสสมอ. เป็นรหัสที่มีการกำหนดตัวอักษรลงในตารางโดยยึดหลักการกำหนดตัวแบบตายตัวเพื่อไว้ใช้ในการแลกเปลี่ยนข้อมูลและข่าวสาร แต่ภาษาไทยที่ใช้ในไมโครคอมพิวเตอร์ส่วนใหญ่ยังต้องการพิมพ์ออกทางเครื่องพิมพ์แบบดอตแมตริกซ์ ซึ่งต้องการอักขระผสมในกรณี 1, 2 และ 3 ระดับ เช่น ค่ ค้ เพื่อลดโอเวอร์เฮดการพิมพ์ (พิมพ์ 3 ครั้งได้ 4 ระดับ) เครื่องพิมพ์แบบดอตแมตริกซ์ยังไม่มีความฉลาดพอที่จะจัดเรียงสระให้เองโดยอัตโนมัติ ดังนั้น รหัสสมอ. จึงมีผู้กำหนดรหัสผสมกันเองโดยชอบของแต่ละคน จึงทำให้การใช้งานไม่เป็นมาตรฐานเดียวกัน รหัสสมอ.น่าจะกำหนดในระดับที่สูงขึ้นมาอีกระดับ คือ แนะนำว่าถ้าจะผสมสระเพื่อพิมพ์หรือแสดงผลควรจะเป็นอย่างไร จะได้เป็นมาตรฐานเดียวกัน และนอกจากนี้ยังน่าจะกำหนดรูปแบบอักขระวิธีเช่น การจัดเรียง ก ุ ้ ง ให้ชัดเจน เพื่อการเรียงอักขระในข้อมูลจะเป็นแบบเดียวกัน

ภาษาไทยมองแบบรูปภาพ : รหัสสองไบต์แนวทางของค่ายตะวันออก

หากพิจารณาการจัดภาษาไทยแบบรูปภาพ โดยรวม 4 ระดับ ลงมาเหลือระดับเดียว ก็จะเห็นภาษาไทยมีลักษณะแบบตัวอักษรภาษาทางค่าย HANZI ได้ เช่น

กุ้ เป็น 1 ภาพ

และดูจากการใช้รหัส 2 ไบต์ใน

กบร้อง เป็น Conclusion

ลักษณะเช่นนี้อาจกล่าวได้ว่า ถ้าฝนตกเมื่อไรกบต้องร้อง แต่กบมีสิทธิ์ร้องได้ ถ้าฝนไม่ตกก็ได้ เพราะกบอาจนึกอยากจะร้องด้วยเหตุอื่นก็ได้ ด้วยวิธีการของ propositional logic เราแทนเหตุการณ์ด้วยสัญลักษณ์ได้ดังนี้

$P \rightarrow Q$

ซึ่งกล่าวโดยสรุปได้ดังนี้

ถ้า P เป็น F ดังนั้น $P \rightarrow Q$ เป็น T
ถ้า Q เป็น T ดังนั้น $P \rightarrow Q$ เป็น T
ทางเดียวที่ $P \rightarrow Q$ จะเป็น F คือ P เป็น T และ Q เป็น F

ลักษณะของ propositional logic

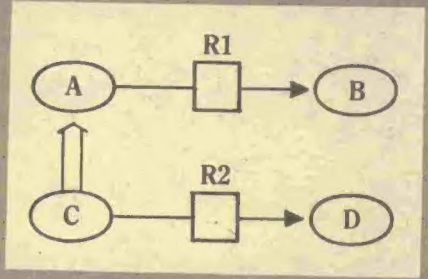
โดยการแทนสัญลักษณ์เข้ากับเหตุผลและข้อความนี้เองทำให้เราสามารถแทนภาษาธรรมชาติในรูปแบบของคณิตศาสตร์ลอจิกได้ โดยมีคุณสมบัติของการแทนดังนี้

1. ใช้สัญลักษณ์ 'A' ถึง 'Z' แทน

2. กำหนดความถูกต้องทางหลักภาษาซึ่งต้องพิสูจน์ได้ ในกรณีนี้เราเรียกว่า well formed.

3. ถ้า P และ Q เป็น well formed ใด ๆ ผลทางลอจิกจะเป็นดังรูป A

การเขียนประโยค NL ด้วย proposition จะสามารถใช้แทนรูปแบบของความสัมพันธ์ในเน็ตเวอร์กได้ (semantic net) ต้องลองดูตัวอย่าง ↓



ในกรณีนี้เราจะมองความสัมพันธ์ของสิ่งต่าง ๆ และพัฒนาขึ้นมาเป็น Predicate logic ซึ่งเป็นโครงสร้างพื้นฐานของโปรล็อก ในที่สุดจากตัวอย่างที่ยกมาในรูปเป็นวิธีการสื่อความหมายและกฎเกณฑ์การถ่ายทอด ซึ่งในวงกลม

~P	~P	เป็น T	เมื่อ P เป็น F
P∧Q	P∧Q	เป็น T	เมื่อทั้ง P และ Q เป็น T
P∨Q	P∨Q	เป็น F	เมื่อทั้ง P และ Q เป็น F
P→Q	P→Q	เป็น T	เมื่อ P เป็น F หรือ Q เป็น T
P↔Q	P↔Q	เป็น T	เมื่อ P และ Q มีค่าเหมือนกัน
P+Q	P+Q	เป็น T	เมื่อ P และ Q มีค่าตรงข้าม
P|Q	P|Q	เป็น F	เมื่อทั้ง P และ Q เป็น T
P↓Q	P↓Q	เป็น T	เมื่อทั้ง P และ Q เป็น F

รูป A คณิตศาสตร์ลอจิกของ P และ Q ในกรณีต่าง ๆ

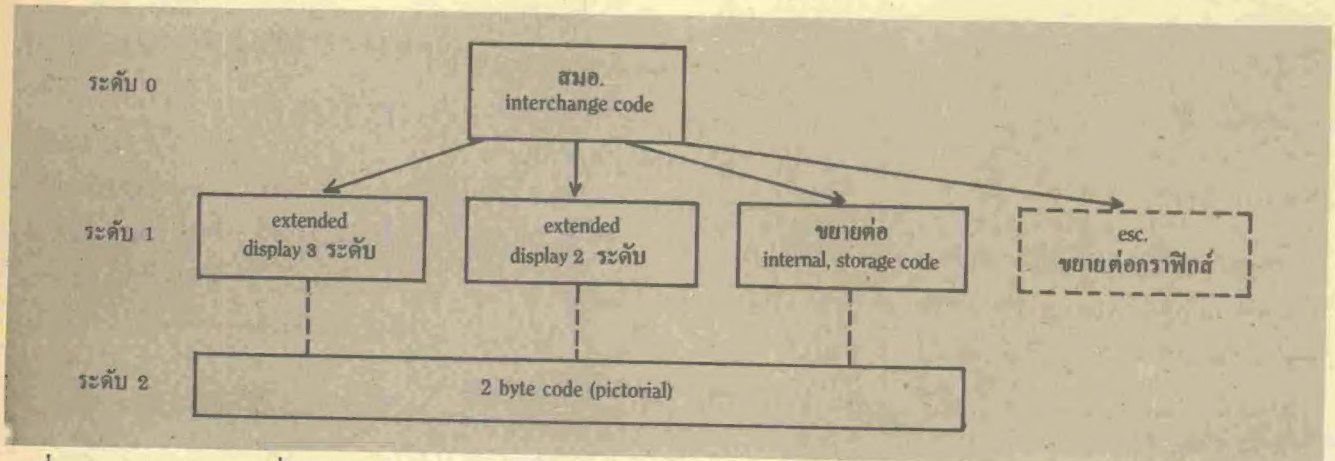
รูปแบบของภาษาทางค่ายตะวันออก เพื่อกำหนดตัวอักขระก็ยิ่งทำให้ภาษาไทยในรหัสมาตรฐานขยายตัวเองเป็นแบบ 2 ไบต์ได้ เช่น ในภาษาญี่ปุ่นมีการส่งรหัส esc เข้าสู่รหัส 2 ไบต์ ได้ เช่น

1B 4B หมายถึง switch on to kanji
1B 48 หมายถึง switch off kanji

ในขณะที่ไม่ใช้ kanji ภาษาญี่ปุ่นจะใช้ภาษาอังกฤษร่วมกับ katakana จะใช้รหัสเพียงไบต์เดียว และเมื่อสวิตช์มาที่ kanji จะใช้รหัส 2 ไบต์ รหัสภาษาไทยก็มีแนวทางการมองแบบ 2 ไบต์ได้เช่นกันกับของ kanji ในภาษาญี่ปุ่นกำหนด JIS 8 ให้เป็นมาตรฐาน มี kanji มาตรฐาน 2,965 ตัว และมี JIS เติมรูปสำหรับ hirakana อีก 453 ตัว และ JIS ครึ่งรูปสำหรับ hirakana 156 ตัว การแสดงผลของภาษาไทยถ้ามองแบบภาษาตะวันออกก็น่าจะมีรูปแบบที่ขยายต่อจาก สมอ.ได้

สมอ.ควรมีหลายระดับ : 1-2 ไบต์ จะออกมาในรูปแบบใด

จากที่กล่าวแล้วว่า รหัสสมอ.เป็นรหัสที่ยังใช้แตกต่างกันได้ เพราะรหัสสมอ. เปิดช่องว่างให้ผู้ใช้ไปใส่สระ



รูปที่ 9 โครงร่างของรหัสที่มีข้อแนะนำหลายระดับ

และมีตัวอักษร คือ A,B,C,D จะเป็น unary predicate เช่น นักร้อง (พุ่มพวง) คำว่า นักร้อง คือ predicate มีอาร์กิวเมนต์เพียงตัวเดียว และถ้าพุ่มพวงเป็นนักร้องจริง จะได้

นักร้อง (พุ่มพวง) = T

แต่ถ้าหากมานะไม่ใช่นักร้องเราจะได้

นักร้อง (มานะ) = F

จากโครงสร้างดังรูป R1 และ R2 เป็น binary predicate ลักษณะของ binary predicate คือ มีอาร์กิวเมนต์ 2 ตัวเช่น

JOHN LOVE MARY

จะเขียนเป็น

LOVE (JOHN, MARY).

ในการแทนความหมายของ R1 และ R2 เราอาจเขียนเป็น

R1 (X,Y).

หรือกล่าวได้ว่า X และ Y เป็นอาร์กิวเมนต์

และเพื่อให้มีการแทนภาษาธรรมชาติได้เด่นชัดขึ้น

$\forall x$ หมายถึง all

$\exists x$ หมายถึง some

ดังนั้นจากรูปภาพเน็ตเวอร์กเราเขียนได้

$(\forall x)$ [C (x) $\rightarrow$ A (x)].

และถ้าเชื่อมโยงกับ R1 จะเขียนได้เป็น

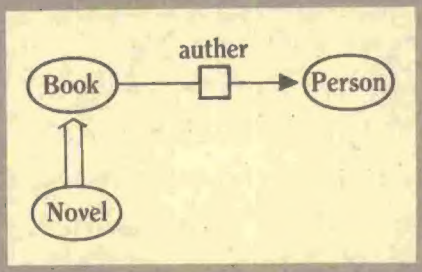
$(\forall x)(\forall y)$ [A(x) $\wedge$ R1(x,y) $\rightarrow$ B(y)].

และถ้าเขียนแทนความรอบรู้ทั้งหมดของเน็ตเวอร์กนั้นจะได้

$(\forall x)$ $(\forall y)$ [C(x) $\wedge$ R1(x,y) $\rightarrow$ B(y)].

$(\forall x)$ $(\forall y)$ [C(x) $\wedge$ R2(x,y) $\rightarrow$ D(y)].

ลองดูตัวอย่างของจริงดู



เราเขียนแทนความรอบรู้นี้ได้ดังนี้

$(\forall x)$ [novel (x) $\rightarrow$ book(x)].

$(\forall x)$ $(\exists y)$ [book(x) $\rightarrow$ author(x,y)].

ด้วยวิธีการทาง predicate logic เราจึงเอามาเขียนแทนประโยค NL ได้ เช่น "คนไทยบางคนชอบอาหารทุกอย่างที่เผ็ด"

ถ้ากำหนดให้ X แทนเซตของคน

Y แทนเซตของอาหาร

เราเขียนประโยค NL ที่ได้ดังนี้

$(\exists x)$ {Thai(x) $\wedge$ $(\forall y)$[food(y) $\wedge$ hot (y) $\wedge$ like(x,y)]].

หรือตัวอย่างข้อความ

"คนขายเนื้อสัตว์ที่ไม่กินเนื้อสัตว์ทุกคนเป็นผู้หญิง"

เขียนประโยค NL นี้ได้ดังนี้

$(\forall x)$ {[$\wedge$(x) $\wedge$ B(x)] $\rightarrow$ F(x)].

โดยในที่นี้เราให้

M(x) แทน x เป็นผู้ชาย

F(x) แทน x เป็นผู้หญิง

V(x) แทน x เป็นคนไม่กินเนื้อสัตว์

B(x) แทน x เป็นคนขายเนื้อสัตว์

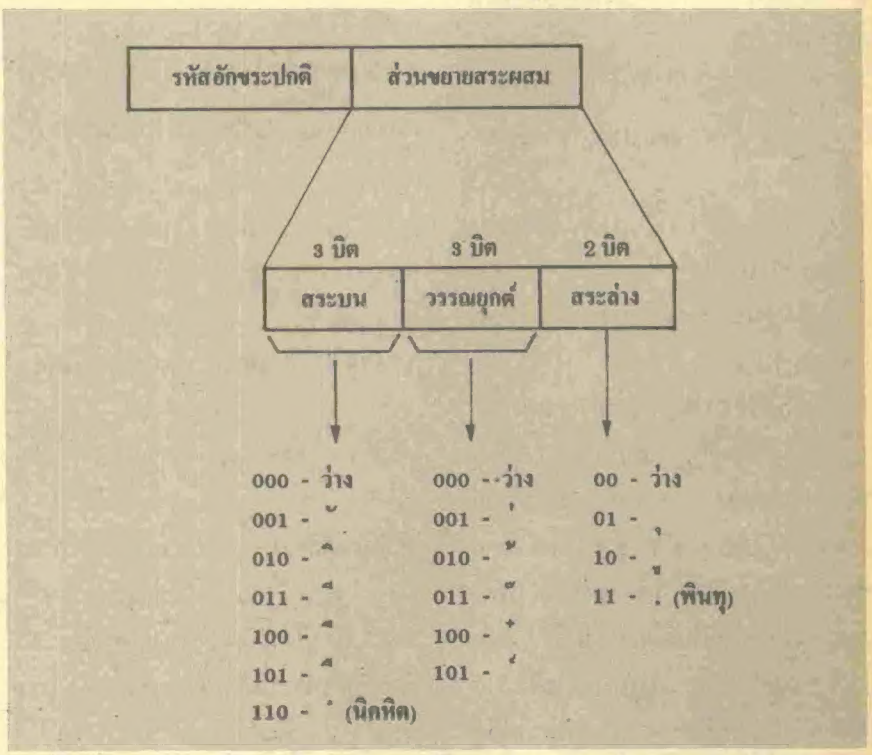
■

ผสมเองตามใจชอบ ความสมบูรณ์ในตัวเองจึงไม่มี เราจึงเห็นว่ารหัส สมอ. ไม่สามารถใช้ร่วมกันในทางปฏิบัติจริงได้ หากกำหนดออกมาเป็นระดับเช่น โครงร่างในรูปที่ 9 ก็จะเป็นทางออกที่สมบูรณ์ได้

2 ไบต์จะเป็นอย่างไร

ในการมองภาษาไทยที่ขยายออกมาเป็น 2 ไบต์นั้นมีทางออกได้หลายรูปแบบ เช่นอาจจะมองคำหนึ่งคำเป็น 2 ไบต์ เหมือนภาษาญี่ปุ่น (kanji) เช่น กิน, ข้าว แต่ในความเห็นของผู้เขียนใคร่อยากเสนอในแนวทางแรกที่มอง 2 ไบต์ แบบ pictorial สำหรับใช้กับเทคโนโลยีทางตะวันออกในอนาคตโดยใช้ 2 ไบต์ ดังรูปที่ 10

สองไบต์ตามแนวความคิดของผู้เขียนที่อยากเสนอแนะสำหรับการใช้ซอฟต์แวร์ทางภาษาในกลุ่ม hanzi นั้นก็เหมือนกับการมองภาษาไทยใน 4 ระดับเป็นรูปภาพ 1 รูป นั่นเอง การพิจารณามาตรฐานของ 2 ไบต์ที่ผู้เขียนยกตัวอย่างนี้ สามารถขยายเชื่อมต่อจากระดับล่าง



รูปที่ 10 รูปแบบการขยายอีก 1 ไบต์ เพื่อมองภาษาไทยแบบ pictorial

ขึ้นมาและเน้นการสวิตช์เข้าสู่รหัส 2 ไบต์ เช่น kanji และที่น่าพิจารณาต่อไปคือโปรแกรมจัดระบบงานรุ่นใหม่เช่น OS/2 สามารถรองรับรหัส 2 ไบต์ได้ หรือแม้แต่โปรแกรมสำเร็จรูปบางโปรแกรม เช่น โลตัส 123 เวอร์ชัน 2J ที่ใช้ในญี่ปุ่นก็ใช้รหัสสองไบต์ได้

NLP V.S. ภาษาศาสตร์

นักภาษาศาสตร์ได้ให้หลักการและรูปแบบและโครงสร้างทางภาษาซึ่งเกี่ยวพันกับการนำมาใช้ในเรื่อง NLP อย่างยิ่ง นักภาษาจะเป็นผู้ที่รู้เกี่ยวข้องกับโครงสร้างและไวยากรณ์ของข้อความ ศึกษาถึงกลไกความสัมพันธ์ของประโยคและความหมาย ตลอดจนพยายามหากฎเกณฑ์หรือโมเดลมาใช้ทางภาษาศาสตร์ นักภาษาศาสตร์จึงศึกษาในแขนงวิชาการทางด้านภาษาต่าง ๆ กันออกไปหลายระดับ เช่น

ระดับการออกเสียง (phonetic และ phonolygy)

ระดับเกี่ยวกับรูปร่างลักษณะของคำและพยางค์ (morphophonolygy and morpholygy)

ระดับวากยสัมพันธ์ (syntax)

ระดับเกี่ยวกับความหมาย (semantics)

ระดับที่เกี่ยวพันกันในข้อความ (pragmatics)

ระดับข้อความและเอกสาร (Text linguistics)

การศึกษาในรูปแบบ formal language จึงหันมาใช้รูปแบบทางด้านคณิตศาสตร์ประยุกต์และนักคอมพิวเตอร์หลายคนได้พยายามใช้หลักการทางคณิตศาสตร์ลอจิกเข้ามาแทนหลักการทางภาษาและที่สำคัญในระยะหลังนี้คือ predicate logic ซึ่งสามารถนำมาใช้ในการแทนกฎ และหลักการทางภาษาศาสตร์ได้ดี มีการพยายามหาวิธีการแทนกฎ เช่น แทนหลักการทางไวยากรณ์ภาษา การแทนความหมาย

NLP จึงเกี่ยวข้องร่วมกับนักภาษาศาสตร์เพื่อหาทางนำเอาหลักการมาใช้ให้เกิดประโยชน์ เช่น นักภาษาต้องการรู้โครงสร้างประโยคที่ซับซ้อน การออกเสียง และความหมาย แต่ในทาง NLP ต้องการหาแบบที่สั้นที่สุดที่จะได้ความหมายครบเพื่อนำไปประมวลผล นักภาษาต้องการเขียนโครงสร้างของความหมายและสิ่งที่สัมพันธ์กัน แต่ทาง NLP ต้องการความเข้าใจในเรื่องความหมาย และสามารถถ่ายทอดต่อไปได้ ความต้องการทาง NLP ยังต้องการประยุกต์ภาษาศาสตร์ไปใช้ในด้านต่าง ๆ อีกด้วย

โครงสร้างทางภาษาศาสตร์ที่เกี่ยวข้องกับ NLP

ในการประมวลผลในรูปแบบภาษาธรรมชาติจะเกี่ยวข้องในหลายระดับ โดยทั่วไปเราจะให้คอมพิวเตอร์วิเคราะห์และเข้าใจในหลักการทางภาษาหลายอย่าง เช่น

การวิเคราะห์ในระดับที่เกี่ยวกับรูปร่างลักษณะของคำและพยางค์ (Morphology)

เป็นระดับที่ต้องพิจารณารูปลักษณะของคำ เช่น การเป็นคำนาม คำกริยา คำคุณศัพท์

นายแดงไปโรงเรียน คำว่า แดงเป็นคำนาม

ขี้ยามีอาการลงแดง คำว่า แดง หรือ ลงแดงเป็นกริยา

หนังสือเล่มสีแดง คำว่า แดง เป็นคำคุณศัพท์

การศึกษาทางลักษณะคำเป็นเรื่องที่สำคัญ บางครั้งก็มีกฎเกณฑ์ตายตัว บางครั้งก็หากฎเกณฑ์ตายตัวไม่ได้ เช่น

child many childen

news new
walks

สิ่งที่สำคัญ คือ ลักษณะคำจะเกี่ยวข้องกับ syntax ของประโยคด้วย เช่น คำว่า "แดง" ดังได้กล่าวแล้ว ลองดูตัวอย่างเพิ่มเติมเป็นภาษาอังกฤษ

John saw a big stone.

This is a stone wall.

การวิเคราะห์ในระดับวากยสัมพันธ์ (syntax)

เป็นระดับที่ศึกษาถึงรูปแบบโครงสร้างของไวยากรณ์ของประโยค ในบางครั้งไวยากรณ์ของภาษาจะสร้างความกำกวมได้ เช่น

old men and women

อาจจะหมายถึง old men and old women หรือ old men and women ก็ได้

ในเรื่องของวากยสัมพันธ์บางครั้งประโยคอาจจะเขียนได้ในรูปแบบหลายอย่างเช่น

The dog bit the man.

The man was bitten by the dog.

คอมพิวเตอร์จะทำการตรวจสอบไวยากรณ์ และจะบอกได้ว่าไวยากรณ์ของประโยคถูกหรือผิดยกตัวอย่างประโยคภาษาไทยที่มีโครงสร้างไวยากรณ์ที่ผิด เช่น "สมชาย ข้าว กิน" เนื่องจากลำดับที่ถูกต้องของประโยคภาษาไทยจะต้องเป็นประธาน กริยา และกรรมเป็น "สมชาย กิน ข้าว" เป็นต้น

การวิเคราะห์เชิงความหมาย (Semantic)

ประโยคที่ถูกวางแบบโดยหลักการทางไวยากรณ์นี้จะต้องมีความหมายแน่นอนอย่างใดอย่างหนึ่ง ในทางภาษาศาสตร์ การออกเสียงเหมือนกันอาจมีความหมายต่างกันก็ได้ เช่น

$bear_1$ = "give birth"

$bear_2$ = "tolerate"

$bear_3$ = "wild animal"

หรือที่เขียนเหมือนกันแต่อ่านออกเสียง

ต่างกันต่างความหมาย เช่น

lead "be the leader"

lead "heavy metal"

การให้ความหมายนี้ทางคอมพิวเตอร์จะต้องวิเคราะห์โครงสร้างทางไวยากรณ์ด้วย เช่น She can not bear children คำว่า bear นี้เป็น verb เนื่องจากประโยคหนึ่ง ๆ อาจจะให้ความหมายที่กำกวมได้ หรือบางประโยคอาจจะมีคำศัพท์หรือรูปแบบโครงสร้างไวยากรณ์ที่ถูกต้องก็จริง แต่อาจจะไม่ให้ความหมายอะไรเลยก็ได้ คอมพิวเตอร์ควรจะมีความสามารถในการแยกแยะความถูกต้องของความหมายได้ด้วย ยกตัวอย่างประโยคภาษาอังกฤษ เช่น "Colorless green ideas sleep juriously" เป็นประโยคที่ถูกไวยากรณ์แต่ก็ไร้ความหมาย

การวิเคราะห์เชิงปฏิบัติ (pragmatic analysis)

หลังจากที่คอมพิวเตอร์เข้าใจโครงสร้างและความหมายของประโยคแล้ว ขั้นตอนที่สำคัญต่อมาก็คือ การตีความและลงมือกระทำตามความหมายหรือความเข้าใจนั้นต่อไป เช่น เมื่อคอมพิวเตอร์ได้รับคำถามว่า "ขณะนี้ กี่ โมง แล้ว" คอมพิวเตอร์ตีความหมายของประโยคแล้ว สิ่งที่ต้องลงมือปฏิบัติก็คือ พิมพ์ผลลัพธ์ของเวลาขณะนี้ลงบนจอภาพ เป็นต้น

โดยปกติการวิเคราะห์ทั้ง 4 แบบนี้ อาจจะต้องกระทำไปพร้อม ๆ กัน หรือมีการสัมพันธ์กันอยู่ จนบางครั้งอาจจะแยกไม่ออกก็ได้ การวิเคราะห์ทั้งหมดนี้เป็นหัวใจของการประมวลผลภาษาธรรมชาติ นั่นเอง

รูปที่ 11 โครงสร้างประโยคในภาษาอังกฤษ

```
Sentence      -->  noun-phrase,verb-phrase
noun-phrase   -->  determiner,noun
noun-pharse   -->  noun
verb-phrase   -->  verb,noun-phrase
verb-phrase   -->  verb
determiner    -->  "the"
determiner    -->  "a"
verb          -->  "likes"
verb          -->  "hates"
noun          -->  "John"
noun          -->  "Mary"
```

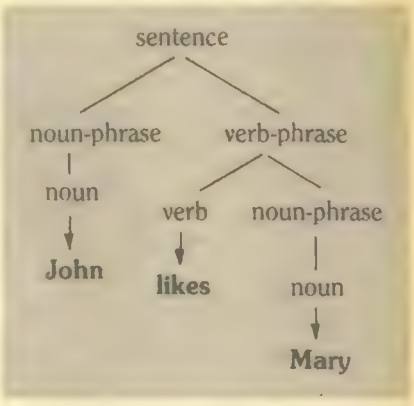
ตัวอย่างการวิเคราะห์โครงสร้างไวยากรณ์ (Syntactical Parsing of NL)

การวิเคราะห์โครงสร้างไวยากรณ์ของภาษาธรรมชาติ ก็คือ การที่รับประโยคในภาษามนุษย์ (ในที่นี้จะแสดงการวิเคราะห์ประโยคภาษาอังกฤษ) และทำการตรวจสอบโครงสร้างที่เรียกว่า "Parse Tree" ถ้าประโยคที่ถูกป้อนเข้ามาไม่ตรงกับโครงสร้างไวยากรณ์ของภาษาแบบใดแบบหนึ่ง แสดงว่าประโยคนั้นมีโครงสร้างไวยากรณ์ที่ผิด ดังนั้นโครงสร้างจึงมีความสำคัญขึ้นตรงกับลักษณะของภาษาใดภาษาหนึ่งโดยตรง ยกตัวอย่าง โครงสร้างประโยคภาษาอังกฤษดังรูปที่ 11

โดยลักษณะการกำหนดโครงสร้างไวยากรณ์ดังกล่าว จะทำให้คอมพิวเตอร์ทำความเข้าใจหรือสามารถ "parse tree" ของประโยคต่อไปนี้ได้

John likes Mary.

และโครงสร้างของ "parse tree" จะเป็นดังนี้



โครงสร้างทางไวยากรณ์เหล่านี้สามารถเขียนในรูปของ (facts) และ "กฎ" (rules) ในภาษาโปรล็อกได้ ดังแสดงในโปรแกรมรูป 12 ขอให้สังเกตว่าเราสามารถสร้าง หรือเพิ่มเติมโครงสร้างไวยากรณ์ของภาษา และรวมทั้งคำศัพท์ต่าง ๆ เข้าไปในระบบได้โดยง่ายในรูปของโปรล็อก

รูปที่ 12 ตัวอย่างของ Syntactical Parsing

```
X sentence if
          APPEND (Y Z X) and
          Y noun-phrase and
          Z verb-phrase

X noun-phrase if
          X modified-name
(X!Y) noun-phrase if
```

รูปที่ 12 (ต่อ)

```
            X determiner and
            Y modified-noun

(X) verb-phrase if
            X intransitive-verb
(X!Y) verb-phrase if
            X transitive-verb and
            Y noun-phrase

(X) modified-name if
            X name
(X Y) modified-name if
            X adjective and
            Y name

(X) modified-noun if
            X noun
(X Y) modified-noun if
            X adjective and
            Y noun

big adjective
hairy adjective

Mary name
John name

kisses transitive-verb
kicks transitive-verb

smiles intransitive-verb
jumps intransitive-verb

the determiner
a determiner
every determiner

man noun
woman noun
dog noun
 confirm( (John kicks a big dog) sentence)
```

วิเคราะห์โครงสร้างไวยากรณ์

```
   ==> Yes , I can confirm (John kicks a big dog) sentence  how

To deduce
      (John kicks a big dog) sentence
I used the rule
X sentence if
          APPEND (Y Z X) and
          Y noun-phrase and
          Z verb-phrase

I can show
      1       APPEND ((John) (kicks a big dog) (John kicks a big dog))
      2       (John) noun-phrase
      3       (kicks a big dog) verb-phrase
       Type a number
  3
```

ผู้ใช้ถามเหตุผลในลักษณะ why หรือ how ได้

ถามคำถามลึกลงไปในแต่ละขั้นได้

```
To deduce
      (kicks a big dog) verb-phrase
I used the rule
(X!Y) verb-phrase if
          X transitive-verb and
          Y noun-phrase

I can show
      1       kicks transitive-verb
```

รูปที่ 12 (ต่อ)

```
      2      (a big dog) noun-phrase
       Type a number
  2 ←— ถามเหตุผลของ noun-phrase
To deduce
      (a big dog) noun-phrase
I used the rule
(X Y) noun-phrase if
          X determiner and
          Y modified-noun

I can show
      1      a determiner
      2      (big dog) modified-noun
       Type a number
  2

To deduce
      (big dog) modified-noun
I used the rule
(X Y) modified-noun if
          X adjective and
          Y noun

I can show
      1      big adjective
      2      dog noun
```

ลักษณะของไวยากรณ์ที่กำหนดอาจจะมีข้อบกพร่องบางประการ ที่จะต้องปรับปรุงเพื่อที่จะทำให้คอมพิวเตอร์ทำการตรวจสอบ ความสัมพันธ์ระหว่างประธานและกริยา กล่าวคือถ้าประธานเป็นเอกพจน์บุรุษที่ 3 กริยาที่ตามมาจะต้องเติม s หรือ es ด้วย ประโยคที่คอมพิวเตอร์จะตรวจสอบความถูกต้องได้ ยกตัวอย่างเช่น

The boys like the girls.

The boy likes Mary.

Bill likes the girls.

The boy walks.

ในขณะเดียวกันคอมพิวเตอร์ต้องวิเคราะห์ได้ว่าประโยคต่อไปนี้ไม่ถูกต้อง

The boys likes the girls.

A boy like the girls.

ยิ่งกว่านั้นตามหลักไวยากรณ์ของภาษาอังกฤษนั้นเราทราบว่า กริยาของประโยคจะแบ่งเป็น 2 ชนิดใหญ่ ๆ ด้วยกัน คือ "อกรรมกริยา" และ "สกรรมกริยา" (กริยาที่ไม่ต้องมีกรรม และกริยาที่ต้องมีกรรมมารับ ตามลำดับ) ดังนั้นประโยคที่ผิดไวยากรณ์ที่คอมพิวเตอร์ตรวจสอบได้ ยกตัวอย่าง เช่น ประโยค

The boy walks the girl.

ดังนั้น หลังจากการเพิ่มกฎทางด้านการตรวจสอบลักษณะของกริยาดังกล่าวแล้ว คอมพิวเตอร์จะยอมรับตัวอย่างของประโยคต่อไปนี้ได้ถูกต้อง เช่น

The boys like the girl.

The boys walk.

และจะบอกได้ว่าประโยคต่อไปนี้ไม่ถูกต้อง

The boys like.

The boy walk the girls.

ตัวอย่างการตรวจสอบประโยคในเชิงความหมาย (Semantic Analysis)

หลังจากที่คอมพิวเตอร์มีความสามารถในการตรวจสอบเชิงไวยากรณ์ไม่ว่าจะเป็นลำดับของประโยค ความสัมพันธ์ของประธานและกริยา รวมทั้งมีการตรวจสอบ อกรรมกริยา และ สกรรมกริยาแล้ว การวิเคราะห์ในเชิงความหมายก็เป็นสิ่งที่สำคัญในลำดับต่อมาด้วย ขอให้พิจารณาประโยคต่อไปนี้

The boys eat the stones.

The stones eat the boys.

จะสังเกตเห็นว่าประโยคทั้ง 2 ข้างบนนี้ เป็นประโยคที่ถูกต้องตามหลักไวยากรณ์ดังกล่าวมาแล้วทั้ง 2 ประโยค แต่ในเชิงของความหมายแล้วเรากล่าวว่าประโยคทั้ง 2 ประโยคจะเป็นไปไม่ได้ เช่น เราจะกล่าวว่า "หิน กิน เด็ก" ได้อย่างไรในเมื่อหินเป็นสิ่งไม่มีชีวิต จึงกระทำกริยา "กิน" ไม่ได้ เป็นต้น

ดังนั้นเพื่อจะเพิ่มคุณสมบัติในการวิเคราะห์เชิงความหมายให้กับคอมพิวเตอร์นั้น เราจะต้องกำหนดกฎความรอบรู้ในเรื่องภาษาเพิ่มเติมเข้าไปอีก แต่ในที่นี้จะเห็นว่า สิ่งที่เพิ่มเติมเข้าไปจะเกี่ยวกับความรู้ต่าง ๆ ที่เกี่ยวข้องรอบ ๆ ตัวด้วย (World knowledge)

รูปที่ 13 โปรแกรมตรวจสอบ transitive, Singular, plural พร้อมทั้งตรวจสอบเหตุผลทาง semantic ด้วย

```
(X (the|Y) Y) determiner
(singular (a|X) X) determiner

(animal singular (man|X) X) noun
(animal plural (men|X) X) noun
(animal singular (dog|X) X) noun
(animal plural (dogs|X) X) noun
(abstract singular (idea|X) X) noun
(abstract plural (ideas|X) X) noun
(vegetable singular (apple|X) X) noun
(vegetable plural (apples|X) X) noun
(mineral singular (stone|X) X) noun
(mineral plural (stones|X) X) noun

((subject animal) transitive (object vegetable) singular (bites|X) X) verb
((subject animal) transitive (object vegetable) plural (bite|X) X) verb
((subject animal) transitive X singular (likes|Y) Y) verb
((subject animal) transitive X plural (like|Y) Y) verb
((subject animal) intransitive X singular (walks|Y) Y) verb
((subject animal) intransitive X plural (walk|Y) Y) verb

X sentence Y if
          (Z x X y) noun_phrase and
          (Z x y Y) verb_phrase

(X Y Z x) noun_phrase if
          (Y Z y) determiner and
          (X Y y x) noun

(X Y Z x) verb_phrase if
          ((subject X) transitive (object y) Y Z z) verb and
          (y X1 z x) noun_phrase
(X Y Z x) verb_phrase if
          ((subject X) intransitive y Y Z x) verb
 confirm( (the man likes the stones) sentence () )
```

คำสั่งตรวจสอบความถูกต้อง

```
   ==> Yes , I can confirm (the man likes the stones) sentence ()  why
```

ถามเหตุผล

```
To deduce
      (the man likes the stones) sentence ()
I used the rule
X sentence Y if
          (Z x X y) noun_phrase and
          (Z x y Y) verb_phrase

I can show
      1       (animal singular (the man likes the stones) (likes the stones)) nou
n_phrase
      2       (animal singular (likes the stones) ()) verb_phrase
       Type a number
  1
```

เหตุผลของ noun-phrase ในรายละเอียด คืออะไร

```
To deduce
      (animal singular (the man likes the stones) (likes the stones)) noun_phras
e
I used the rule
(X Y Z x) noun_phrase if
          (Y Z y) determiner and
          (X Y y x) noun

I can show
      1       (singular (the man likes the stones) (man likes the stones)) determ
iner
      2       (animal singular (man likes the stones) (likes the stones)) noun
       Type a number
  2
I know that (animal singular (man|X) X) noun
  stop

Execution terminated
&.
```

ขอให้พิจารณาโปรแกรมโปรล็อกในรูปที่ 13 ที่ได้บรรจุการตรวจสอบทั้งด้านไวยากรณ์และความหมายลงไปด้วย จะสังเกตเห็นว่าลักษณะคำศัพท์ต่าง ๆ จะมีข้อมูลที่เกี่ยวข้องบางประการเพิ่มเติมลงไป เช่น คำนามอาจจะต้องมีการแยกแยะชัดเจนลงไป เช่น คำนามนั้นเป็นชนิด abstract, animal, vegetable หรือ mineral เป็นต้น ส่วนคำกริยาก็จะต้องมีการกำหนดกลุ่มของประธานที่จะกระทำกริยาตัวนี้ได้ และถ้าเป็นกริยาแบบ "สกรรมกริยา" ก็จะต้องมีการกำหนดกลุ่มหรือชนิดของกรรมที่จะมารับด้วย

ลักษณะข้อมูลของนามแสดงในรูปที่ 14 และของกริยาแสดงในรูปที่ 15

จากการแทนความรอบรู้ในด้านไวยากรณ์และความหมายดังกล่าวนี้ จะทำให้คอมพิวเตอร์สามารถรับประโยคที่ถูกต้องในทั้งสองแง่ ดังตัวอย่างต่อไปนี้

The men bite the apples.

The dogs bite the apples.

The men like the dog.

The men likes the apple.

The men walks.

ในทำนองเดียวจะสามารถบอกได้ว่า ประโยคต่อไปนี้ไม่ถูกต้องตามความหมาย (แม้ว่าจะถูกต้องตามไวยากรณ์) เช่น

The apple walks.

The apples bite the idea.

The idea bites the stone.

The men bites the stone.

```
noun(animal,singular) --> [man]
noun(animal,plural)   --> [men]
noun(animal,singular) --> [dog]
noun(animal,plural)   --> [dogs]
noun(abstract,singular) ->[idea]
noun(abstract,plural) --> [ideas]
noun(vegetable,singular)->[apple]
noun(vegetable,plural) -->[apples]
noun(mineral,singular) -->[stone]
noun(mineral,plural)  --> [stones]
```

← รูปที่ 14

↓ รูปที่ 15

```
verb(subject(animal),transitive,object(vegetable),singular)
--> [bites]
verb(subject(animal),transitive,object(vegetable),plural)
--> [bite]
```

รูปที่ 16 โปรแกรม Debugging & Correcting English syntax

```
((verb_phrase (X Y Z x))
  (verb ((subject X) transitive (object y) Y Z z))
  (noun_phrase (y x1 z x)))
((verb_phrase (X Y Z x))
  (verb ((subject X) intransitive y Y Z x)))
((noun_phrase (X Y Z x))
  (determiner (Y Z y))
  (noun (X Y y x)))
((sentence X Y)
  (noun_phrase (Z x X y))
  (verb_phrase (Z x y Y))
  (PP the sentence is correct ))
((sentence X Y)
  (noun_phrase (Z x X y))
  (nonverb_phrase (Z x y Y))
```

โปรแกรมการตรวจสอบและ แก้ไขประโยคภาษาอังกฤษ (Debugging and Correction of English Syntax)

นอกจากคอมพิวเตอร์จะมีความสามารถตรวจสอบว่าประโยคในภาษาอังกฤษประโยคใดถูกต้องหรือผิดไปจากหลักไวยากรณ์ หรือหลักของความหมายแล้ว ในกรณีที่ประโยคไม่ถูกต้องนั้น คอมพิวเตอร์ยังสามารถบอก หรืออธิบายได้ว่าความผิดพลาดนั้นเกิดเนื่องจากอะไร เช่น ผิดหลักไวยากรณ์ (ประธานเป็นเอกพจน์บุรุษที่สามกริยาต้องเติม s หรือ es เป็นต้น) หรือผิดความหมาย (Semantics) คอมพิวเตอร์ยังบอกได้อีกด้วยว่า โปรแกรมนั้นที่ถูกต้องตามหลักไวยากรณ์ควรจะเป็นอย่างไร ขอให้พิจารณาจากการทำงานของโปรแกรมได้ดังต่อไปนี้ ↓

```
the ?((run))
the man walk.
the incorrect verb is (w a l k)
the correct verb is (w a l k s)
&?((run))
the men walks.
the incorrect verb is (w a l k s)
the correct verb is (w a l k)
&?((run))
the man walks.
the sentence is correct
&?((run))
the man walks the apple.
the sentence is meaningless
&?((run))
the apple bites the man.
the sentence is meaningless
&?((run ))
the dog bites the apple.
the sentence is correct
&.
```

รูปที่ 16 (ต่อ)

```
    (PP the sentence is meaningless))
  ((sentence X Y)
    (noun_phrase (Z x X y))
    (notverb_phrase (Z x y Y))
    (PP the sentence is meaningless))
  ((sentence X Y)
    (noun_phrase (Z x X y))
    (cut y Y1))
  ((determiner (X (the!Y) Y)))
  ((determiner (singular (a!X) X)))
  ((verb ((subject animal) transitive (object vegetable) singular (bites!X) X)))
  ((verb ((subject animal) transitive (object vegetable) plural (bite!X) X)))
  ((verb ((subject animal) transitive X singular (likes!Y) Y)))
  ((verb ((subject animal) transitive X plural (like!Y) Y)))
  ((verb ((subject animal) intransitive X singular (walks!Y) Y)))
  ((verb ((subject animal) intransitive X plural (walk!Y) Y)))
  ((notverb ((subject X) transitive (object Y) singular (bites!Z) Z)))
  ((notverb ((subject X) transitive (object Y) plural (bite!Z) Z)))
  ((notverb ((subject X) intransitive Y singular (walks!Z) Z)))
  ((notverb ((subject X) intransitive Y plural (walk!Z) Z)))
  ((noun (animal singular (man!X) X)))
  ((noun (animal plural (men!X) X)))
  ((noun (animal singular (dog!X) X)))
  ((noun (animal plural (dogs!X) X)))
  ((noun (abstract singular (idea!X) X)))
  ((noun (abstract plural (ideas!X) X)))
  ((noun (vegetable singular (apple!X) X)))
  ((noun (vegetable plural (apples!X) X)))
  ((noun (mineral singular (stone!X) X)))
  ((noun (mineral plural (stones!X) X)))
  ((cut (x!X) z)
    (STRINGOF Z x)
    (PP the incorrect verb is Z)
    (correct Z z)
    (PP the correct verb is z))
  ((pack X Y Y)
    (lastword (X)))
  ((pack X Y Z)
    (APPEND Y (X) x)
    (R y)
    (pack y x Z))
  ((reverse (X!Y) (X))
    (EQ Y ()))
  ((reverse (X!Y) Z)
    (reverse Y x)
    (APPEND x (X) Z))
  ((APPEND () X X))
  ((APPEND (X!Y) Z (X!x))
    (APPEND Y Z x))
  ((tail-s (e i!X) ("y"!X)))
  ((tail-s (e h t!X) (e h t!X)))
  ((tail-s (e h!X) (h!X)))
  ((tail-s (e "x"!X) ("x"!X)))
  ((tail-s (e s s!X) (s s!X)))
  ((tail-s (e s!X) (s!X)))
  ((tail-s (e "x"!X) ("x"!X)))
```

รูปที่ 16 (ต่อ)

```
((tail-s (e v!X) (f!X)))
((tail-s (e v!X) (e f!X)))
((tail-s (e o!X) (o!X)))
((tail-s X X))
((lastword (.)))
((correct Z z)
  (reverse Z (s!Y))
  (tail-s Y y)
  (reverse y z))
((correct Z z)
  (reverse Z Y)
  (tail- Y y)
  (reverse y z))
((tail- ("y"!X) (s e i!X)))
((tail- ( h t!X) (s e h t!X)))
((tail- ( h!X) (s e h!X)))
((tail- ( "x"!X) (s e "x"!X)))
((tail- ( s s!X) (s e s s!X)))
((tail- ( s!X) (s e s!X)))
((tail- ( "x"!X) (s e "x"!X)))
((tail- ( f!X) (s e v!X)))
((tail- ( e f!X) (s e v!X)))
((tail- ( o!X) (s e o!X)))
((tail- X (s!X)))
((run)
  (read x)
  (sentence x () ))
((nonverb_phrase (X Y Z x))
  (verb ((subject X) intransitive (object y) Y Z z))
  (noun_phrase (y X1 z x)))
((nonverb_phrase (X Y Z x))
  (verb ((subject X) transitive y Y Z x))
  (noun_phrase (y X1 z x)))
((notverb_phrase (X Y Z x))
  (notverb ((subject X) transitive (object y) Y Z z))
  (noun_phrase (y X1 z x)))
((notverb_phrase (X Y Z x))
  (notverb ((subject X) intransitive y Y Z x))
  (noun_phrase (y X1 z x)))
((read X)
  RCLEAR
  (R Y)
  (EQ Z ())
  (pack Y Z X)
  (/* read a sentence from keyboard , X is the return value .*))
```

บทสรุป

เนื่องจากภาษาที่มนุษย์ใช้ในการพูดคุย ในการอ่านและเขียน มีความอ่อนตัวหรือกฎข้อยกเว้นที่ไม่ตายตัวมากมาย บางครั้งต้องใช้หลักการหรือสามัญสำนึกบางอย่างเพื่อช่วยในการตัดสินใจ อาจจะกล่าวได้ ความสามารถที่มนุษย์มีในการเข้าใจภาษานั้นเป็นตัวอย่างที่ดีที่สุดอันหนึ่งในการแสดงออกถึงความฉลาดที่มนุษย์มีที่แตกต่างจากเครื่องจักรหรือสิ่งมีชีวิตอื่น ๆ ดังนั้นการพัฒนาและวิจัยหรือที่จะทำให้คอมพิวเตอร์มีความสามารถดังกล่าวจึงเป็นเรื่องยาก แต่ก็เป็นเรื่องที่จะต้องให้ความสนใจในการสร้างเครื่องมือและหลักวิชาการต่าง ๆ เสริม ซึ่งแน่นอนว่าจะให้ประโยชน์ต่อการใช้งานที่มากขึ้น และใช้ในทางปฏิบัติมากขึ้นต่อไปในอนาคตอันใกล้